IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: **Martin et al.**

Serial No.: **09/784,694**

Filed: **February 15, 2001**

For: **Method and System for Specifying a Cache Policy for Caching Web Pages Which Include Dynamic Content**

Group Art Unit: **2157**

Examiner: **El Chanti, Hussein A.**

Attorney Docket No.: **RSW920010011US1**

36736
PATENT TRADEMARK OFFICE
CUSTOMER NUMBER

Certificate of Transmission Under 37 C.F.R. § 1.8(a)
I hereby certify this correspondence is being transmitted via facsimile to the Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, facsimile number (703) 872-9306 on June 23, 2005.
By: Jennifer Pitcher

TRANSMITTAL DOCUMENT

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

RECEIVED
OIPE/IAP
JUN 24 2005

Sir:
ENCLOSED HEREWITH:

- Appeal Brief (37 C.F.R. 41.37).

A fee of $500.00 is required for filing an Appeal Brief. Please charge this fee to IBM Corporation Deposit Account No. 09-0461. No additional fees are believed to be necessary. If, however, any additional fees are required, I authorize the Commissioner to charge these fees which may be required to IBM Corporation Deposit Account No. 09-0461. No extension of time is believed to be necessary. If, however, an extension of time is required, the extension is requested, and I authorize the Commissioner to charge any fees for this extension to IBM Corporation Deposit Account No. 09-0461.

Respectfully submitted,

Brian Owens
Registration No. 55,517
PATENT AGENT FOR APPLICANTS
Duke W. Yee
Registration No. 34,285
ATTORNEY FOR APPLICANTS
YEE & ASSOCIATES, P.C.
P.O. Box 802333
Dallas, Texas 75380
(972) 385-8777

Yee & Associates, P.C.

4100 Alpha Road
Suite 1100
Dallas, Texas 75244

Main No. (972) 385-8777
Facsimile (972) 385-7766

Facsimile Cover Sheet

To: Commissioner for Patents for **Examiner Hussein A. El Chanti** **Group Art Unit 2157**	Facsimile No.: **703/872-9306**
From: Jennifer Pilcher Legal Assistant to Brian Owens	No. of Pages Including Cover Sheet: **28**
Message: Transmitted herewith: • Transmittal Document; and • Appeal Brief.	
Re: Application No.: **09/784,694** Attorney Docket No: **RSW920010011US1**	
Date: Thursday, June 23, 2005	
Please contact us at (972) 385-8777 if you do not receive all pages indicated above or experience any difficulty in receiving this facsimile.	*This Facsimile is intended only for the use of the addressee and, if the addressee is a client or their agent, contains privileged and confidential information. If you are not the intended recipient of this facsimile, you have received this facsimile inadvertently and in error. Any review, dissemination, distribution, or copying is strictly prohibited. If you received this facsimile in error, please notify us by telephone and return the facsimile to us immediately.*

PLEASE CONFIRM RECEIPT OF THIS TRANSMISSION BY FAXING A CONFIRMATION TO 972-385-7766.

Docket No. RSW920010011US1 *PATENT*

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: **Martin et al.**	§	
	§	Group Art Unit: **2157**
Serial No. **09/784,694**	§	
	§	Examiner: **El Chanti, Hussein A.**
Filed: **February 15, 2001**	§	
	§	
For: **Method and System for Specifying a Cache Policy for Caching Web Pages Which Include Dynamic Content**	§	Attorney Docket No.: **RSW920010011US1**

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Certificate of Transmission Under 37 C.F.R. § 1.8(a)
I hereby certify this correspondence is being transmitted via facsimile to the Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, facsimile number (703) 872-9306 on June 23, 2005.

By: Jennifer Pilcher
Jennifer Pilcher

APPEAL BRIEF (37 C.F.R. 41.37)

This brief is in furtherance of the Notice of Appeal, filed in this case on April 25, 2005.

The fees required under § 41.20(B)(2), and any required petition for extension of time for filing this brief and fees therefore, are dealt with in the accompanying TRANSMITTAL OF APPEAL BRIEF.

06/24/2005 HLE333 00000034 090461 09784694
01 FC:1402 500.00 DA

REAL PARTY IN INTEREST

The real party in interest in this appeal is the following party: International Business Machines Corporation

RELATED APPEALS AND INTERFERENCES

With respect to other appeals or interferences that will directly affect, or be directly affected by, or have a bearing on the Board's decision in the pending appeal, there are no such appeals or interferences.

STATUS OF CLAIMS

A. TOTAL NUMBER OF CLAIMS IN APPLICATION

Claims in the application are: 1-42.

B. STATUS OF ALL THE CLAIMS IN APPLICATION

1. Claims canceled: NONE
2. Claims withdrawn from consideration but not canceled: NONE
3. Claims pending: 1-42
4. Claims allowed: NONE
5. Claims rejected: 1-42
6. Claims objected to: NONE

C. CLAIMS ON APPEAL

The claims on appeal are: 1-42

STATUS OF AMENDMENTS

A Response to the Final Office Action was filed on March 23, 2005; however, no amendments were made to the claims in the Response. Claims 1-42 on appeal herein are as finally rejected in the Final Office Action mailed January 25, 2005.

SUMMARY OF CLAIMED SUBJECT MATTER

A. Independent Claims 1, 15 and 29:

The subject matter of claim 1 is directed toward a method, system, and computer program for specifying a cache policy for caching pages which include dynamic content. A system in which the invention may be implemented is illustrated in **Figure 4** and is described beginning on page 9, line 15 and extending to page 12, line 6. A user is permitted to request one of the pages to be displayed. See page 10, line 19-26. The page includes multiple fragments. See page 11, lines 20-28. An application is executed that includes multiple servlets, each of which is executed to present a different one of the multiple fragments. See page 6, lines 19-20. The servlets are unchanged by the caching policy. See page 12, lines 1-6. See page 12, lines 1-6. Each one of the servlets is executed to present a different one of the fragments. See page 6, lines 20-21. Caching of the page fragments is processed separately from the execution of the application and its servlets. See page 12, lines 1-3. One of multiple cache options is specified based on an update rate of content of one of the multiple servlets. The options include static caching, dynamic caching or no caching. See **Figure 5**, page 19, line 5 through page 20, line 17. Content that is updated dynamically is cached using static caching, dynamic caching, or no caching.

B. Dependent Claims 6, 20, 34

Claim 6, 20 and 34 are dependent on the method, system and computer program elements of claims 1, 15 and 29 respectively. Claim 6 further recites the steps of receiving a request to display one of the plurality of fragments (see page 9, lines 17-21, **step 802 Figure 8**), determining one of plurality of servlets associated with one of the plurality of fragments (see page 10, lines 7-26 and page 11, lines 14-28, and executing one of the plurality of servlets wherein the execution of one of the plurality of servlets generates a displayable output (see page 24 line 28 through page 25 line 6, steps **822**, **824** and **826** in **Figure 8**).

GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

The grounds of rejection on appeal are as follows:

I. Claims 1-42 are rejected under 35 U.S.C. §103(a) as being allegedly unpatentable over Hawes (U.S. Patent No. 6,094,662).

ARGUMENT

I. 35 U.S.C. §103(a), Alleged Obviousness Claims 1-42.

The Final Office Action rejects claims 1-42 under 35 U.S.C. § 103(a) as being unpatentable over Hawes. (U.S. Patent No. 6,094,662) (hereinafter "*Hawes*"). This rejection should be reversed.

IA. Claims 1-5, 15-19, 29-33

Hawes teaches an apparatus and method for loading and reloading HTML pages having cacheable and non-cacheable portions. The HTML portions of the page are marked as non-cacheable which allows the web page to be retrieved without retrieving cacheable graphics images. A refresh function allows a time stamp of non-cached portions to be compared with the time stamps of the web server so that when the web server has a more current time stamp, a browser displaying the web page can be updated with the non-cached portion. Additionally, a timer or refresh button on the web page or browser may be employed to update the non-cached portion of the web page.

In rejecting the claims as being obvious over *Hawes*, the Examiner states the following:

> As to claims 1, 15 and 29. Hawes teaches a method, system, and program respectively in a data processing system for specifying a cache policy for caching pages which include dynamic content, said method, system and program comprising the steps of:
>
> permitting a user to request one of said pages to be displayed, said one of said pages including a plurality of fragments (see col. 4 lines 45-47, user retrieves web pages);
>
> executing an application which includes a plurality of HTML portions, each one of said plurality of HTML portions being executed to present a different one of said plurality of fragments, each one of said plurality of HTML portions being unchanged by said caching policy (see col. 4 lines 48-52, browser separates cacheable and non-cacheable portions); and
>
> processing caching of said one of said pages separately from said application; and
>
> specifying one of a plurality of different caching options for one of said plurality of HTML portions based on an update rate of content of said one of said plurality of HTML portions, said plurality of different caching options including either static caching, dynamic caching or no caching content that is updated dynamically being cached either static caching, dynamic caching or no caching

(see col. 5 lines 13-36, browser dynamically and periodically updates the cacheable portion).

Hawes does not explicitly teach the HTML portions are servlets. Official Notice is taken as evident by Microsoft computer Dictionary 5th edition that it would have been obvious for one of the ordinary skill in the art at the time of the invention to modify *Hawes* by using servlets because doing so would make the browser execute quickly and thereby reducing system overhead.

Final Office Action dated March 25, 2005 pages 2 and 3.

Claim 1 is a representative claim in this group and states as follows:

1. A method in a data processing system for specifying a cache policy for caching pages which include dynamic content, said method comprising the steps of:

permitting a user to request one of said pages to be displayed, said one of said pages including a plurality of fragments;

executing an application which includes a plurality of servlets, each one of said plurality of servlets being executed to present a different one of said plurality of fragments, each one of said plurality of servlets being unchanged by said caching policy;

processing caching of said one of said pages separately from said application; and

specifying one of a plurality of different caching options for one of said plurality of servlets based on an update rate of content of said one of said plurality of servlets, said plurality of different caching options including either static caching, dynamic caching or no caching, content that is updated dynamically being cached using either static caching, dynamic caching, or no caching.

Claim 1 recites an application that includes multiple servlets. Each servlet is executed to presents a different one of the multiple fragments of a page. *Hawes* does not suggest using servlets and particularly does not suggest processing caching of the page separate from the application. Page 12, lines 1-6.

The Examiner alleges this argument is ineffective by stating the following in the Advisory Action:

Applicant's arguments were fully considered but are not persuasive. In remarks, applicant argues A) Hawes does not disclose processing caching pages separately from the application.

In response Hawes teaches a system and method for caching webpages by checking HTML portions and determining whether the portion are "cacheable" or "non-cacheable". The determination as to whether the portions are "cacheable" or "non-cacheable" are made using the portions characteristics and not based on the application that requested the application (see col. 6, lines 15-42) and therefore

> Hawes meets the scope of the claimed limitation "processing caching of said one of said pages separately from said application".

Advisory Action dated April 4, 2005 cover page.

This statement is contradicted by specification and claims of *Hawes*. Particularly, the single apparatus claim indicative of a "system" in *Hawes* that claims in part:

> 3. An apparatus that updates a document retrieved from a node of a distributed network to a client, comprising:
>
> a browser that retrieves the document from the node over the distributed network and separates the document into at least one cacheable portion and at least one non-cacheable portion, the at least one cacheable portion stored in- cache memory of the client and the at least one non-cacheable portion stored in non-cache memory of the client, the browser comprising:
>
> a refresh status button that compares each at least one non-cacheable portion of the document stored in the non-cache memory with a corresponding non-cacheable portion of the document stored at the node;
>
> wherein, for each non-cacheable portion of the document stored in the non-cache memory that is different from the corresponding non-cacheable portion of the document stored at the node, the browser retrieves the corresponding non-cacheable portion of the document stored at the node and stores the retrieved non-cacheable portion in the non-cache memory in place of the corresponding non-cacheable portion previously stored in the non-cacheable memory." (emphasis added)

Hawes col. 8, lines 25-48.

Clearly the application or browser in this case is processing the cacheable and non-cacheable portions of the page by separating, retrieving and storing them. *Hawes* further notes in that:

> In the preferred embodiment, the browser **180** accesses the web page URL, and compares the non-cached portion with the corresponding portion of the web page **212**. If a change is detected, the browser **180** requests the non-cacheable portion of the web page **212** to be downloaded to the client 110 and displayed on display **140**. While retrieving the non-cacheable portion of the web page 212 from the web site **210**, the browser retrieves the cached portion of the web page **212** from cache **184** for displaying on display. **140**

See *Hawes* col. 5, lines 13-37.

The "portions characteristics" described by the examiner are used by the browser in processing the caching. The portions characteristics are not used independent of the browser to process the cached elements as suggested.

Hawes explicitly teaches a browser that presents the page and that also performs the caching distinguishable from the processing step of claim1. The browser retrieves a page. The browser separates the page into cacheable portions and non-cacheable portions. See column 4, lines 45-53. The browser then stores the cacheable portions in a cache. See column 4, lines 45-53, column 5, lines 53-56, and column 6, lines 32-33. The browser retrieves the non-cached portion from the web site and the cached portion from the cache for displaying on the display. See column 5, lines 22-25. When a page is to be refreshed, the browser reloads the cached portion from the cache. See column 5, lines 53-56 and column 6, line 49-51.

Thus, according to *Hawes*, the browser presents a page and processes the caches. In *Hawes*, the same "application", i.e. the browser, both presents the page and processes the caching. In *Hawes*, the processing of the caching is not done separately from the application that presented the page as specifically claimed in the processing element of the independent claims.

Appellants teach an application that includes servlets that present the fragments of a page. Processing of the caching is not done, however, by this application. *Hawes* teaches away from the features claimed by Appellants by teaching a browser that both presents a page and does the caching as well. According to Appellants' claims, processing of the caching is done separately from the application that included the servlets that presented the fragments of the page.

Claims 2-5 depend from and further restrict claim 1 and are also not unpatentable over *Hawes*, at least by virtue of their dependency. Therefore, Appellants respectfully requests that the rejection of claims 1-5, 15-19, and 29-33 under 35 U.S.C. §103(a) not be sustained.

IB. Dependent Claims 6-14, 20-28, 34-42

Dependent claim 6 depends from claim 1 further narrowing the scope of claim 1 which recites a method for specifying a cache policy for caching pages which include dynamic content. The Final Office Action states:

> As to claims 6, 20 and 34, Hawes teaches the method, system and program according to claims 1, 15 and 29 respectively, wherein the step of processing caching of said one of said pages further comprises the steps of: receiving a request to display one of said plurality of fragments; determining one of plurality of HTML portions associated with said one of said plurality of fragments; and executing said one of said plurality of HTML portions, wherein said execution of said one of said plurality of HTML portions generates a displayable output (see col. 5 lines 14-36).

Final Office Action dated March 25, 2005 pages 2 and 3.

Claim 6 is a representative claim in this group and reads as follows:

> 6. The method according to claim 1, wherein the step of processing caching of said one of said pages further comprises the steps of:
> receiving a request to display one of said plurality of fragments;
> determining one of plurality of servlets associated with said one of said plurality of fragments; and
> executing said one of said plurality of servlets, wherein said execution of said one of said plurality of servlets generates a displayable output.

Hawes makes no reference to the determining step of the present invention wherein servlets are linked to fragments so that there output can be efficiently displayed. In *Hawes*, the non-cached portion of a web page is downloaded jointly as a portion and not in multiple fragments that are executed using servlets. *Hawes* states "If a change is detected, the browser **180** requests the non-cacheable portion of the web page **212** to be downloaded to the client **110** and displayed on the display **140**." See col. 5, line 20-22. Further, the language cited by the Examiner gives no indicative of a determining step or an execution step. The *Hawes* reference specifically cited provides no determining step in which one of multiple servlets is associated with one of multiple fragments. This is because servlets are not used by *Hawes* and the separation and caching is done by the browser rather than separate from the application as claimed by Appellants. *Hawes* is distinguishable because in the preferred embodiment the browser both caches, retrieves, and displays the non-cached portion of the web page without executing servlets or their substantial equivalent. As a result, Appellants' claim 1 embodies a method different from that suggested by *Hawes*.

Appellants respectfully requests that the rejection of claims 6-14, 20-28, 34-42 under 35 U.S.C. §103(a) not be sustained.

Brian D. Owens
Reg. No. 55,517
YEE & ASSOCIATES, P.C.
PO Box 802333
Dallas, TX 75380
(972) 385-8777

CLAIMS APPENDIX

The text of the claims involved in the appeal are:

1. A method in a data processing system for specifying a cache policy for caching pages which include dynamic content, said method comprising the steps of:

permitting a user to request one of said pages to be displayed, said one of said pages including a plurality of fragments;

executing an application which includes a plurality of servlets, each one of said plurality of servlets being executed to present a different one of said plurality of fragments, each one of said plurality of servlets being unchanged by said caching policy;

processing caching of said one of said pages separately from said application; and

specifying one of a plurality of different caching options for one of said plurality of servlets based on an update rate of content of said one of said plurality of servlets, said plurality of different caching options including either static caching, dynamic caching or no caching, content that is updated dynamically being cached using either static caching, dynamic caching, or no caching.

2. The method according to claim 1, further comprising the steps of processing caching of each of said plurality of fragments separately from said application.

3. The method according to claim 1, wherein the step of processing caching further comprises the steps of:

responding to internal cache requests; and

responding to external cache requests.

4. The method according to claim 1, further comprising the step of processing caching of said one of said pages within an application server included within said computer system.

5. The method according to claim 4, further comprising the steps of:

responding to internal cache requests, said internal cache requests being generated within said application server; and

responding to external cache requests, said external cache requests being generated outside said application server.

6. The method according to claim 1, wherein the step of processing caching of said one of said pages further comprises the steps of:

receiving a request to display one of said plurality of fragments;

determining one of plurality of servlets associated with said one of said plurality of fragments; and

executing said one of said plurality of servlets, wherein said execution of said one of said plurality of servlets generates a displayable output.

7. The method according to claim 6, further comprising the steps of:

in response to a first request to display said one of said plurality of fragments, creating a cache entry including said output;

creating a cache entry identifier for identifying said cache entry utilizing a user identifier which identifies said user and caching options specified for said one of said plurality of servlets.

8. The method according to claim 7, further comprising the steps of:

creating said one of said plurality of servlets;

specifying said cache options for said one of said plurality of servlets; and

creating a servlet element for said servlet utilizing a servlet identifier and an indication of said specified cache options, wherein said servlet element is associated with said servlet.

9. The method according to claim 8, further comprising the steps of:

storing said servlet; and

storing said specification of said servlet options with said servlet.

10. The method according to claim 8, further comprising the steps of:

receiving a request to display said servlet element;

determining whether any cache entry is identified by said cache identifier;

in response to a determination that no cache entry is identified by said cache identifier:

retrieving said servlet associated with said servlet element;

providing said user identifier to said servlet;

executing said servlet utilizing said user identifier generating an output;

storing said output as a cache entry;

identifying said cache entry utilizing said cache identifier; and

returning said cache entry to said user, wherein said output is displayed.

11. The method according to claim 8, further comprising the steps of:

receiving a request to display said servlet element;

determining whether any cache entry is identified by said cache identifier;

in response to a determination that a cache entry exists which is identified by said cache identifier, returning said cache entry to said user, wherein said output is displayed.

12. The method according to claim 6, further comprising the step of outputting said cache entry, wherein said one of said plurality of fragments is displayed.

13. The method according to claim 6, further comprising the step of in response to subsequent requests to display said one of said plurality of fragments, retrieving said cache entry utilizing said cache identifier.

14. The method according to claim 13, further comprising the step of outputting said cache entry, wherein said one of said plurality of fragments is displayed.

15. A data processing system for specifying a cache policy for caching pages which include dynamic content, comprising:

said data processing system for executing an application which includes a plurality of servlets, each one of said plurality of servlets being executed to present a different one of a plurality of fragments included within a page, each one of said plurality of servlets being unchanged by said caching policy;

said data processing system for processing caching of said one of said pages separately from said application; and

one of a plurality of different caching options specified for one of said plurality of servlets based on an update rate of content of said one of said plurality of servlets, said plurality of different caching options including either static caching, dynamic caching or no caching, content that is updated dynamically being cached using either static caching, dynamic caching, or no caching.

16. The system according to claim 15, further comprising said data processing system for processing caching of each of said plurality of fragments separately from said application.

17. The system according to claim 15, further comprising:

said data processing system for responding to internal cache requests; and

said data processing system for responding to external cache requests.

18. The system according to claim 15, further comprising said data processing system for processing caching of said one of said pages within an application server included within said computer system.

19. The system according to claim 18, further comprising:

said data processing system for responding to internal cache requests, said internal cache requests being generated within said application server; and

said data processing system for responding to external cache requests, said external cache requests being generated outside said application server.

20. The system according to claim 15, further comprising:

said data processing system for receiving a request to display one of said plurality of fragments;

said data processing system for determining one of plurality of servlets associated with said one of said plurality of fragments; and

said data processing system for executing said one of said plurality of servlets, wherein said execution of said one of said plurality of servlets generates a displayable output.

21. The system according to claim 20, further comprising:

said data processing system for in response to a first request to display said one of said plurality of fragments, creating a cache entry including said output;

said data processing system for creating a cache entry identifier for identifying said cache entry utilizing a user identifier which identifies said user and caching options specified for said one of said plurality of servlets.

22. The system according to claim 21, further comprising:

said data processing system for creating said one of said plurality of servlets;

said data processing system for specifying said cache options for said one of said plurality of servlets; and

said data processing system for creating a servlet element for said servlet utilizing a servlet identifier and an indication of said specified cache options, wherein said servlet element is associated with said servlet.

23. The system according to claim 22, further comprising:

said data processing system for storing said servlet; and

said data processing system for storing said specification of said servlet options with said servlet.

24. The system according to claim 22, further comprising:

said data processing system for receiving a request to display said servlet element;

said data processing system for determining whether any cache entry is identified by said cache identifier;

said data processing system in response to a determination that no cache entry is identified by said cache identifier:

for retrieving said servlet associated with said servlet element;

for providing said user identifier to said servlet;

for executing said servlet utilizing said user identifier generating an output;

for storing said output as a cache entry;

for identifying said cache entry utilizing said cache identifier; and

for returning said cache entry to said user, wherein said output is displayed.

25. The system according to claim 22, further comprising:

said data processing system for receiving a request to display said servlet element;

determining whether any cache entry is identified by said cache identifier;

said data processing system for in response to a determination that a cache entry exists which is identified by said cache identifier, returning said cache entry to said user, wherein said output is displayed.

26. The system according to claim 20, further comprising said data processing system for outputting said cache entry, wherein said one of said plurality of fragments is displayed.

27. The system according to claim 20, further comprising said data processing system for in response to subsequent requests to display said one of said plurality of fragments, retrieving said cache entry utilizing said cache identifier.

28. The system according to claim 27, further comprising said data processing system for outputting said cache entry, wherein said one of said plurality of fragments is displayed.

29. A computer program product in a data processing system for specifying a cache policy for caching pages which include dynamic content, said computer program product comprising:

instruction means for permitting a user to request one of said pages to be displayed, said one of said pages including a plurality of fragments;

instruction means for executing an application which includes a plurality of servlets, each one of said plurality of servlets being executed to present a different one of said plurality of fragments, each one of said plurality of servlets being unchanged by said caching policy;

instruction means for processing caching of said one of said pages separately from said application; and

instructions for specifying one of a plurality of different caching options for one of said plurality of servlets based on an update rate of content of said one of said plurality of servlets, said plurality of different caching options including either static caching, dynamic caching or no caching, content that is updated dynamically being cached using either static caching, dynamic caching, or no caching.

30. The product according to claim 29, further comprising instruction means for processing caching of each of said plurality of fragments separately from said application.

31. The product according to claim 29, wherein said instruction means for processing caching further comprises:

instruction means for responding to internal cache requests; and

instruction means for responding to external cache requests.

32. The product according to claim 29, further comprising instruction means for processing caching of said one of said pages within an application server included within said computer system.

33. The product according to claim 32, further comprising:

instruction means for responding to internal cache requests, said internal cache requests being generated within said application server; and

instruction means for responding to external cache requests, said external cache requests being generated outside said application server.

34. The product according to claim 29, wherein said instruction means for processing caching of said one of said pages further comprises:

instruction means for receiving a request to display one of said plurality of fragments;

instruction means for determining one of plurality of servlets associated with said one of said plurality of fragments; and

instruction means for executing said one of said plurality of servlets, wherein said execution of said one of said plurality of servlets generates a displayable output.

35. The product according to claim 34, further comprising:

instruction means for in response to a first request to display said one of said plurality of fragments, creating a cache entry including said output;

instruction means for creating a cache entry identifier for identifying said cache entry utilizing a user identifier which identifies said user and caching options specified for said one of said plurality of servlets.

36 The product according to claim 35, further comprising:

instruction means for creating said one of said plurality of servlets;

instruction means for specifying said cache options for said one of said plurality of servlets; and

instruction means for creating a servlet element for said servlet utilizing a servlet identifier and an indication of said specified cache options, wherein said servlet element is associated with said servlet.

37. The product according to claim 36, further comprising:

instruction means for storing said servlet; and

instruction means for storing said specification of said servlet options with said servlet.

38. The product according to claim 36, further comprising:

instruction means for receiving a request to display said servlet element;

instruction means for determining whether any cache entry is identified by said cache identifier;

instruction means in response to a determination that no cache entry is identified by said cache identifier:

for retrieving said servlet associated with said servlet element;

for providing said user identifier to said servlet;

for executing said servlet utilizing said user identifier generating an output;

for storing said output as a cache entry;

for identifying said cache entry utilizing said cache identifier; and

for returning said cache entry to said user, wherein said output is displayed.

39. The product according to claim 36, further comprising:

instruction means for receiving a request to display said servlet element;

instruction means for determining whether any cache entry is identified by said cache identifier;

instruction means for in response to a determination that a cache entry exists which is identified by said cache identifier, returning said cache entry to said user, wherein said output is displayed.

40. The product according to claim 34, further comprising instruction means for outputting said cache entry, wherein said one of said plurality of fragments is displayed.

41. The product according to claim 34, further comprising instruction means for in response to subsequent requests to display said one of said plurality of fragments, retrieving said cache entry utilizing said cache identifier.

42. The product according to claim 41, further comprising instruction means for outputting said cache entry, wherein said one of said plurality of fragments is displayed.

EVIDENCE APPENDIX

There is no evidence to be presented.

RELATED PROCEEDINGS APPENDIX

There are no related proceedings.